PATENT CLAIMS

1. A processor architecture in which the access to a memory occurs via pointers which refer to objects,

   characterized in that pointers are stored in a pointer area and data is stored in a data area separately from one another in the objects, the pointers containing a memory address of the object to which they refer and the objects being provided with attributes which are stored in the object itself and which describe a length of the pointer area and a length of the data area, and

   the processor provides a register set with separate data and pointer registers, of which the pointer registers are used for the access to objects in the memory.

2. The processor architecture according to Claim 1,

   characterized in that the processor ensures that every pointer contains only either a predefined null value or the memory address of an existing object.

3. The processor architecture according to Claim 1 or 2,

   characterized in that a instruction set having separate instructions for data and pointer operations is used.

4. The processor architecture according to one of Claims 1 through 3,

   characterized in that load and store operations for pointers exclusively load pointers from the pointer areas of the objects into the pointer registers and/or

store the content of pointer registers into the pointer areas of the objects, and

load and store operations for data exclusively load data from the data areas of the objects in data registers and/or store the content of data registers in the data areas of the object.

5. The processor architecture according to one of Claims 1 through 4,

characterized in that a instruction set having an object creation instruction is used, which initializes all pointers in the pointer area of a created object with a null value before the created object may be accessed.

6. The processor architecture according to Claim 5,

characterized in that the object creation instruction may be interrupted and resumed at a later point in time.

7. The processor architecture according to Claim 6,

characterized in that in the event of interruption of the object creation instruction, incompletely initialized objects are created, which are clearly differentiated by the processor from completely initialized objects.

8. The processor architecture according to one of Claims 1 through 7,

characterized in that the processor supports constant objects which are kept in a separate memory area that is exclusively read at program runtime, and

pointers to constant objects are clearly identified by the processor.

9. The processor architecture according to one of Claims 1 through 8,

characterized in that a program stack is used, which is divided into a pointer stack area and a data stack area, a length of the occupied part in each of the two stack areas being indicated by a stack index, which is managed in a data register reserved for this purpose.

10. The processor architecture according to Claim 9,

characterized in that a instruction is used for pushing a pointer onto the pointer stack, which both stores the corresponding pointer onto the pointer stack and also increases the pointer stack index in a non-interruptible way.

11. The processor architecture according to one of Claims 1 through 10,

characterized in that the processor supports static objects which are kept in a separate memory area, which is managed by an operating system, and

pointers to static objects are clearly identified by the processor.

12. The processor architecture according to Claim 9 in connection with Claim 11 or Claim 10 in connection with Claim 11,

characterized in that static objects are used for the program stack and that the attributes contained in the

object describe the length of an actually occupied part of the stack area in the event of an inactive program stack.

13. The processor architecture according to one of Claims 1 through 12,

characterized in that an attribute register is assigned to every pointer register, in which the attributes of the object to which the pointer in the pointer register refers are written.

14. The processor architecture according to Claim 13,

characterized in that a pipeline having an additional pipeline stage is used for loading the attributes.

15. The processor architecture according to Claim 13 or 14,

characterized in that an attribute cache is used.

16. The processor architecture according to one of Claims 1 through 15,

characterized in that an RISC instruction set is used.

17. The processor architecture according to one of Claims 1 through 16,

characterized in that the processor performs automatic garbage collection.

18. A processor in which the processor architecture according to one of the preceding claims is implemented.

19. A device having a main processor (1) in which the processor architecture according to one of Claims 1 through 16 is implemented, and a coprocessor (2), which is implemented to perform automatic garbage collection and is closely coupled to the main processor (1) for efficient synchronization.

20. A use of the processor architecture according to one of Claims 1 through 17 for embedded systems.